WHITE PAPER

A LAYMAN'S GUIDE TO THIN CLIENT COMPUTING

A Layman's Guide to Thin Client Computing

Let's first try to understand what thin client computing is and is not.

- Some journalists have prophesied that thin client computing will be the downfall of client server computing. Not so, the thin client model assumes client server architecture.

- You don't need network PCs to implement a thin client application. Your thin client can be a traditional Web browser, a Java-based client or a Network Computer.

- Thin client computing doesn't reduce the amount of processing that needs to take place; it merely moves it from the client PC to a server, (i.e., you now need a super server).

- You don't need a relational database management system (RDMS) to implement a thin client solution. However, the use of a RDMS is highly recommended because of its capabilities: and in particular, its transaction processing capabilities.

- You don't need to use a browser to implement a thin client application. However, it is highly recommended that you do because a browser provides a 'common' and 'standard' client interface.

- You don't need to use either the CORBA (Common Object Request Broker Architecture) or DCOM (Distributed Component Object Model) models. However, it is highly recommended that you do because of the existing development tools and human experience in the market.

- You can use a variety of languages to build a thin client application. However, you are more than likely to use a selection of C++, J++, VBScript, JavaScript and Dynamic HTML.

- A commitment to a thin client solution assumes a commitment to an object-oriented design and implementation.

Thin client computing is about minimizing the amount of processing that takes place on the client and minimizing the amount of data transmitted between the client and the application server. Ideally, a thin client application should not require the loading of any software whatsoever on the client work-station.

Most thin client designs are based upon multi-tier architecture. That is, a thin client, a business server and a database server. Note that I am not referring to hardware; you could install all tiers on a single machine as you would for example, if building a demonstration on a powerful notebook.

The benefits of multi-tier architecture are well described by Nick Snowdon in his excellent book ***Oracle Programming with Visual Basic***. Nick says:

"The multi-tiered approach was designed to solve as many of the client/server failings as possible.

The kinds of features you will find are as follows:

- *Applications are easier to deploy and keep current with multi-tier architecture. This is especially important when you have to scale your applications for use on the Web.*

- *Business rules in a 'thick' client means that each client has to be altered whenever the business rules change.*

- *By encapsulating all the business rules in the middle tier, any application that needs to access these rules can make use of shared components rather than coding the same rules into every application on the client.*

- *The business layer can ensure the security of data in a standard manner and assure that the applications are as stable as possible.*

- *Scalability issues are covered. Although the multi-tier architecture has some overhead for small applications, the benefits pay off when scaling up because you can easily include more middle-tier hardware."*

An Example

Let me try to illustrate the above principles with a simple example. Let's assume that we are designing a thin client application using Microsoft's DCOM architecture. Our client is a PC running Internet Explorer 3 or 4. Our business server (Web server) is an NT box running Transaction Server and Internet Information Server. Our database server is an NT box running SQL Server. Our development languages are VB 6, VBScript and HTML. These are our three tiers.

Once built, our thin client application works as follows:

1. A user runs a browser and obtains a Web page by specifying its Uniform Resource Locator (URL).

2. A Hypertext Transfer Protocol (HTTP) request is then sent to the appropriate Web server by the browser.

3. The request triggers the execution of a program (Active X DLL) on the Web server and a response to the browser.

4. The HTTP response from the Web server contains the Hypertext Markup Language (HTML) for the specified page.

5. The browser interprets and displays the Web page.

6. The Web page is made 'interactive' by the inclusion of a Client script enclosed with the HTML and written in VBScript.

7. Each of the elements on the Web page (buttons, data entry fields, etc) is represented by an object that may have properties (Dynamic HTML). The client script can interact with these objects (press buttons, enter data in a text field, etc) and process events.

8. The Web page also contains Sever script and is known as an 'Active Server Page'.

9. The Server script is used to generate ODBC commands to write and retrieve data from the SQL database. The Server script may also dynamically construct HTML and/or Client script and direct the Active Server Framework (ASF) to write this to the browser.

The above is a very simple example but even so, it contains most of the elements and processes of any thin client application.

The advantages of thin client computing are essentially the same as those described by Nick Snowdon for multi-tier client server architecture. The major added advantage of thin client computing over 'fat client' multi-tier client server computing is that no software needs to be loaded

or maintained at the work station (client) other than the client operating system and browser, e.g., Windows 98 and Internet Explorer 4. **The cost savings to a large organization are therefore tremendous.**

Thin client computing also releases the user from the tyranny of proprietary LANs and WANs. Given appropriate security, the user can connect to the application from anywhere in the world given a computer, phone line, modem and Internet account.

Of course, the success of thin client computing depends upon the availability of adequate communications resources. You basically have three choices for your communications backbone:

1. The World Wide Web (WWW). You can build your private Intranet using the WWW as the communications resource and ensure privacy and data security by utilizing a secure, encrypted protocol. Using this methodology, your encrypted Web packets are secured with a normal Web packet and sent across the Web. Be warned however, that many experts still have serious reservations about this approach and it would generally not be recommended for sensitive data.

2. If you have the money and resources, you can build and maintain your own communications resource (WAN) and avoid using the WWW. This is the most secure, but most expensive implementation. However, the advent and growing popularity of Virtual Private Networks (VPNs) is changing the cost equation and we are now seeing many smaller companies installing VPNs as a relatively low cost way to implement secure Intranets.

3. You can restrict your system to in-house users and run it across your LAN. This option however, is not suitable for multiple offices and for companies with clients and suppliers who need access to the corporate Intranet.

So, why should your business implement all future applications using the thin client, multi-tier model?

The technical benefits are described earlier; the business benefits are that you can enjoy a significantly superior service at a significantly lower cost.

Is there a downside? Yes, there is a downside. Thin client computing is relatively new as are the methodologies (DCOM, CORBA) and tools to implement it. There are very few technicians with the skills and experience required on the market. It requires a significant change in mind-set from both your business analysts and technicians in the design and implementation phases. It is complex, far more so than traditional fat client computing because it requires object-oriented design and programming and the mastering of new technologies.

If you wish to be successful in thin client computing you will need to invest in your staff through training and reference material and you will need to invest in new hardware and software. In my mind however, the benefits far outweigh the costs. The thin client multi-tier paradigm is the only future for all enterprise-wide, mission critical applications.

Written by Frank McKenna, CEO Knowledgeone Corporation
2002

